

## **AMENDMENTS TO THE SPECIFICATION**

Please amend the specification as indicated below. The language being added is underlined (“  ”) and the language being deleted contains a strikethrough (“”) or is enclosed by double brackets (“[[ ]]”).

Please substitute the following paragraph for the current Abstract:

Described are various embodiments for scheduling threads for a multi-tasking operating system. In accordance with some embodiments, a rate-based scheduling algorithm is incorporated to provide a flexible set of Quality of Service guarantees controlling the allocation of CPU to a thread, together with a mechanism to provide latency control for hard-real-time software. Additionally, the present invention is particularly suitable for use in systems where essentially all software runs under the control of the scheduler, from “interrupt handler” functions, through multimedia applications to simple “console” functions and non-real-time tasks. In an additional embodiment, the system and method of the present invention also actively limits the CPU time allocated to a given software thread to defend against malicious or accidental denial of service attacks.

Please amend the paragraph beginning on page 7, line 5 as follows:

[0023] Whether a thread transitions to the running or runnable state depends upon the thread scheduling scheme established for the system. The details of this scheduling scheme are set forth in additional detail below. Further, threads may transition between the running and runnable states at the discretion of the thread scheduler. However, once threads are out of the inactive state, they may not transition[[s]] back to the inactive state except through a direct action of the thread itself. This can only happen if the thread is currently in a running state. The thread

scheduler does not have the ability to transition threads to the inactive state.